Region-based Mixture of Gaussians Modelling for Foreground Detection in Dynamic Scenes

Sriram Varadarajan[a,*], Paul Miller[a], Huiyu Zhou[a]

[a] *The Centre for Secure Information Technologies (CSIT), Queen's University Belfast, Belfast BT3 9DT, United Kingdom*

Abstract

One of the most widely used techniques in computer vision for foreground detection is to model each background pixel as a Mixture of Gaussians (MoG). While this is effective for a static camera with a fixed or a slowly varying background, it fails to handle any fast, dynamic movement in the background. In this paper, we propose a generalised framework, called region-based MoG (RMoG), that takes into consideration neighbouring pixels while generating the model of the observed scene. The model equations are derived from Expectation Maximisation theory for batch mode, and stochastic approximation is used for online mode updates. We evaluate our region-based approach against ten sequences containing dynamic backgrounds, and show that the region-based approach provides a performance improvement over the traditional single pixel MoG. For feature and region sizes that are equal, the effect of increasing the learning rate is to reduce both true and false positives. Comparison with four state-of-the art approaches shows that

*Corresponding author

Email addresses: svaradarajan01@qub.ac.uk (Sriram Varadarajan), p.miller@qub.ac.uk (Paul Miller), h.zhou@ecit.qub.ac.uk (Huiyu Zhou)

RMoG outperforms the others in reducing false positives whilst still maintaining reasonable foreground definition. Lastly, using the ChangeDetection (CDNet 2014) benchmark, we evaluated RMoG against numerous surveillance scenes and found it to amongst the leading performers for dynamic background scenes, whilst providing comparable performance for other commonly occurring surveillance scenes.
Keywords: Region based Modelling, Moving Object Detection, Mixture of Gaussians, Dynamic Background Subtraction, Expectation Maximisation

1. Introduction

Foreground detection is often the first step in the automated analysis of video surveillance data. Whilst there has been a significant amount of research activity into this problem [1, 2, 3], with the advent of many ingenious approaches, it is fair to say that the robust performance required for real world applications is still some way off. Outdoor camera deployments are particularly problematic, due to the movement of the background scene itself. Examples include trees swaying in the wind, waves rippling through water and fire. Of particular interest to us, is the moving background seen through the windows of public transport platforms, such as buses and trains, when they are in motion.

One of the reasons for the poor practical performance in outdoor scenes is that most of the current approaches assume a stationary background and are also pixel-based. The latter is a consequence of real-time performance constraints; however, recent advances in hardware mean that real-time region-based approaches are now feasible [4]. Many current approaches involve con-

structing a probabilistic background model for each pixel individually. The landmark paper in this regard was [5], with much of the current research based on this approach [6]. In order to handle global changes in background scenes due to illumination variation, they modelled each image pixel as a Mixture of Gaussians (MoG) whose parameters were learnt using a heuristic online k-means approximation. Whilst this type of approach works well for a static or a slowly changing background, it does not handle fast, dynamic changes in the background due to the motion described above. This is mainly because it assumes that pixels can be modelled independently. However, in all these cases, it can be noted that the background is not static, rather it is complex, and tends to periodically repeat over time in a neighbourhood region. These types of background changes are called dynamic textures and they exhibit certain spatio-temporal stationarity properties [7]. By exploiting these properties, we extend the traditional pixel-based mixture modelling approaches over neighbourhood regions.

In this paper, we present a framework for region-based mixture modelling. In section 2, we extensively review other background modelling algorithms. Our region-based modelling concept is briefly introduced in section 3. In section 4, we derive update equations using the Expectation Maximisation (EM) algorithm. We then describe our region-based foreground detection algorithm in section 5. Section 6 describes its evaluation and discusses the results obtained. Finally, we conclude the paper in section 7.

2. Related Work

The system proposed by Stauffer and Grimson [5] is the de facto standard for probabilistic modelling of background pixels based on Gaussian mixtures. Since then there have been various other pixel based methods [6] that have improved upon this method. KaewTraKulPong and Bowden [8] suggested the use of different update equations during the learning and detection phases of the tracker in order to increase its speed and accuracy. Zivkovic [9] proposed using a Dirichlet prior to dynamically estimate the number of Gaussians. Wang and Miller were the first to formally derive the update equations using EM, and derived new update equations by employing regularisation [10]. While these methods model the pixels using Gaussian parameters, there are also non-parametric modelling techniques[11, 12, 13] that use Kernel Density Estimation to estimate the probability density value of the pixels at each time instant. However, all these methods are per-pixel approaches and hence do not successfully solve the problem of dynamic backgrounds.

By considering a region, along with the pixel temporal values, dynamic textures can be modelled with greater fidelity. Many researchers have indeed tried to use the region context to model dynamic backgrounds in scenes. Some notable approaches include Sheikh and Shah [14] who used a kernel based approach that took into account neighbouring pixel locations whilst modelling the background of a particular pixel. However, they had to maintain a history of frames based on the learning rate, resulting in a high dimensionality problem. Jodoin et al [15] proposed a combined spatial and temporal framework that assumed the spatial variations over a region were the same as the temporal variation of an individual pixel. However, the au-

thors themselves state that this assumption does not always hold true, such as in the case of a blinking light.

A background model based on spatio-temporal textures was proposed by Yumiba et al [16] in which they handled local and global changes in the background using a "Space-Time Patch" of gradients. However, their method could only be used for detection of motion as opposed to foreground segmentation. Zhang et al [17] used a non-parametric spatial-temporal model for modelling pixels, over a set of frames. Since their approach requires maintaining pixel samples from a series of previous frames, it poses a memory requirement issue as in the case of most non-parametric techniques.

Dalley et al [18] proposed a heuristic generalisation of the MoG model to handle dynamic textures. They allowed a pixel model to be generated from any/all pixels in its neighbourhood region using a window based operation. Four different update methods were suggested based on different combinations of hard and soft classification decisions. However, because different Gaussians could independently affect the same pixel at the same time, they used multiple simultaneous measurements. This increased the computational complexity of the algorithm.

Dickinson et al. [19] also made use of the spatial relationship between pixels while modelling homogeneous regions in a scene as a MoG distribution. While they have used separate Gaussians for spatial and colour distributions, we allow our colour distribution to be modelled over a region. Their model exploits spatial homogeneity in scenes, whereas our approach models dynamic textures varying over a region.

In [20], we gave a brief overview of a region-based Mixture of Gaussians

(RMoG) algorithm without any rigorous theoretical treatment and presented some preliminary results. In the current work, we show the detailed derivation of both batch and online mode update equations for the region based mixture modelling approach by using EM theory in section 4. Furthermore, in this work we generalise the modelling framework by introducing features of different sizes, whereas in the previous work the modelling was restricted to a feature size of one pixel. Lastly, we have significantly increased the experimental evaluation to ensure greater rigour and improved confidence in our results.

While Gaussian mixture modelling has widely been used in the literature, there are two major issues that we attempt to address with our work. The first one is that most statistical modelling approaches do not address the issue of pixel dependency, i.e. they consider pixels to be independent while modelling a scene. While this is done for purposes of computational efficiency, it does not actually reflect the real scenarios we are interested in, such as onboard public transport vehicles. In our case the background contains static and moving background regions which generate dynamic textures. By developing a framework for region based background modelling, we address this issue by taking pixel dependency into consideration. Our framework also treats the traditional MoG algorithm as a special case when the region is considered to be degenerate. The second issue that we address is that many approaches are heuristic in nature and lack rigour in theoretical treatment. By deriving the updating equations from EM theory, we base our approach on a strong theoretical foundation. We also provide update equations for batch processing thus enabling this approach to be used in other

machine learning applications as well. Also, by including different feature sizes in our framework, we bring pixel-level and block-level modelling under one roof for the first time, to the best of our knowledge.

3. Region-based Mixture Modelling

In the case of standard mixture modelling, the intensity distribution at each pixel is represented by an MoG model learnt over time. Since the scene background is assumed to be constant over long periods of time, the background mixtures will have high weights and low variances. The density function of the pixel is then given by

$$p(y_{i,j}|\theta) \propto \sum_{h=1}^{H} \omega_h \mathcal{N}(y_{i,j}|\mu_h, \Sigma_h) \tag{1}$$

where y is the intensity of the pixel at location (i, j), the parameter set of the mixtures $\theta = \{\omega, \mu, \Sigma\}$ includes the weight, mean and covariance matrix for each mixture, h is the index for each mixture, and H is the number of mixtures at that pixel location.

While this is a very good assumption, it is not very effective for fast, dynamic background movements, e.g., leaves swaying in the wind. This type of movement can also occur due to camera jitter, a common occurrence on pole mounted video surveillance cameras. Furthermore, dynamic backgrounds, such as smoke or water, tend to repeat patterns that are localised to a small region. Thus, due to this movement, one can assume that the intensity captured at a pixel could come from imaging a point in the scene anywhere over this local region, rather than from the same scene point over time. This gives rise to the idea of modelling pixels as a mixture of distributions spread over

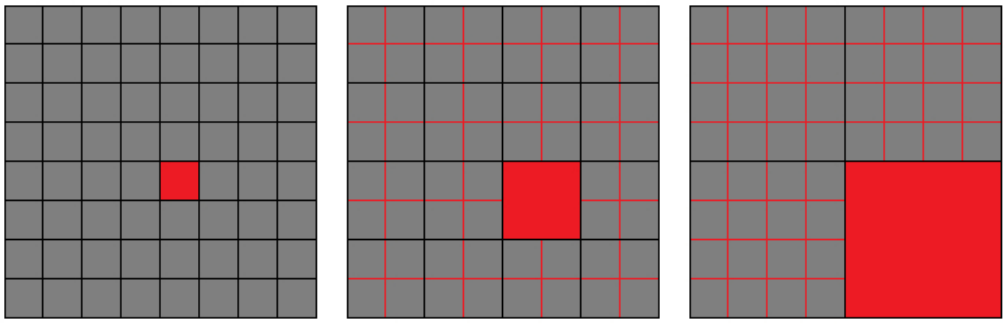


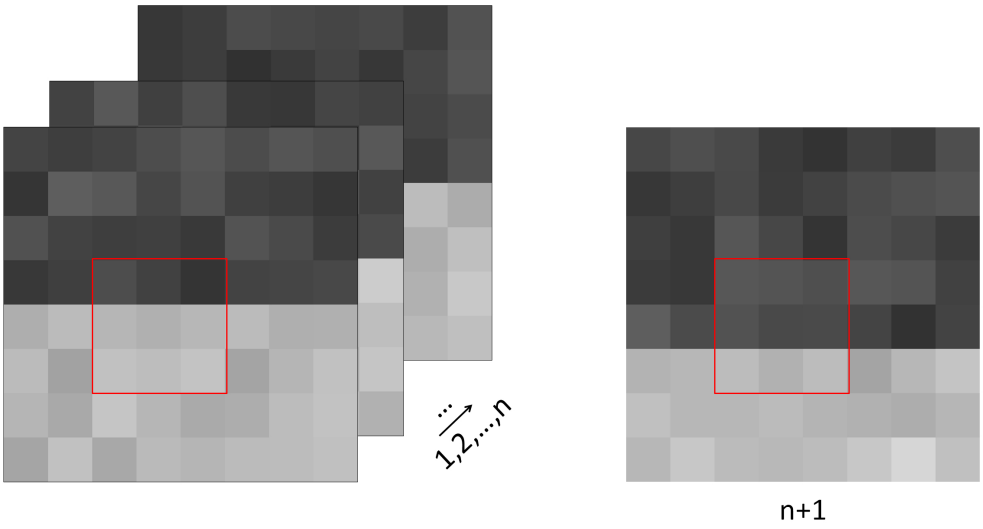
Figure 1: Left: r=8 (1×1 feature block); Middle: r=8 (2×2 feature block); Right: r=8 (4×4 feature block) (Red lines - Pixel boundaries, Black lines - Feature block boundaries, Red region - Reference feature block in the region)

a square neighbourhood $\mathcal{R}$, of size $r \times r$, such that the probability is given by

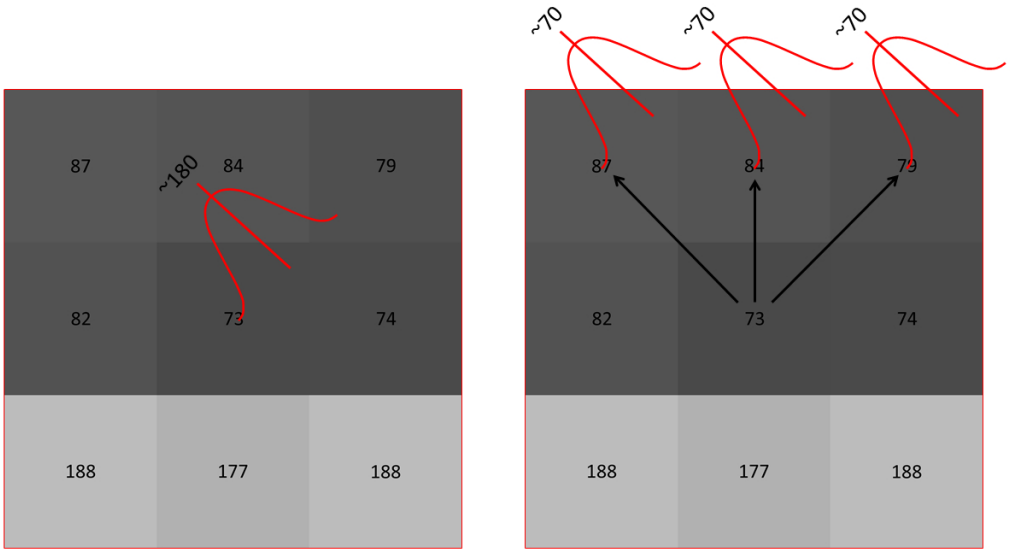
$$p(\boldsymbol{y}_{i,j}|\theta) \propto \sum_{q \in \mathcal{R}_{i,j}} \sum_{h=1}^{H} \omega_{qh} \mathcal{N}(\boldsymbol{y}_{i,j}|\mu_{qh}, \Sigma_{qh}) \tag{2}$$

Here, an additional subscript q is included to clearly show where the chosen mixture component is located in the neighbourhood $\mathcal{R}$. Comparing equation (2) with equation (1), the first summation becomes degenerate when the neighbourhood is reduced to the pixel location alone. Also, please note that $\boldsymbol{y}$ is now in bold type indicating a feature vector. As we have increased the region size r, this means we no longer have to limit ourselves to a feature size of 1×1 pixel. For example, for an 8×8 region, we can select block features of size $f \times f$ where f can be 2, 4 and 8, in addition to the conventional feature of size 1×1, Fig. 1. Similarly, for a region of size 4×4 we can select block feature sizes of 2×2 and 4×4. And so on for different region sizes.

The difference between the two approaches in relation to background modelling is highlighted by a simple illustration in Fig. 2. Consider a se-



(a) Sequence of n images with downward shift at time $n+1$



(b) Left: Standard MoG, Right: Region-based MoG

Figure 2: Illustration of Region-based Mixture Modelling

quence of n images of size 8×8 pixels. Each image can viewed as made up of two areas of different pixel intensities, dark (∼70) and light (∼180) . At time instant n+1, there is a small movement in the scene that causes the edge to shift down by one row, Fig. 2(a). Now, let us look at the central pixel in the 3×3 region given by the red box. In the case of standard mixture modelling, this pixel will be considered a foreground pixel as the model at that location is learnt solely from data due to the light area, see Fig. 2(b) Left. However, in the case of region modelling, this pixel will still be considered a background pixel as the model in the 3×3 region are learnt from data in both the light and dark areas, see Fig. 2(b) Right. Hence, the pixel will be able to be classified as belonging to one of the mixtures from the top row in the 3×3 region. We formalise this idea in the following section by deriving update equations for this type of modelling.

4. Expectation Maximisation for Region-based Mixture of Gaussians

Consider a series of N images $Y = \{Y_1, Y_2, ..., Y_n, ..., Y_N\}$ and their associated hidden variables $Z = \{Z_1, Z_2, ..., Z_n, ..., Z_N\}$ where n is the index of the current image. For each image and latent variable of size (I, J), we can define separate 2-D rectangular lattices $S = \{i, j : 1 \leq i \leq I, 1 \leq j \leq J\}$ and $U = \{k, l : 1 \leq k \leq I, 1 \leq l \leq J\}$ such that $Y_n = \{\boldsymbol{y}_{n,i,j} : i, j \in S\}$ and $Z_n = \{\boldsymbol{z}_{n,k,l} : k, l \in U\}$, where $\boldsymbol{z}_{n,k,l}$ is a H-dimensional binary vector having a value for each mixture h. The neighbourhood space of $\boldsymbol{z}_{n,k,l}$ is given by $\{\boldsymbol{z}_{n,k,l} : k, l \in \mathcal{R}_{k,l}\}$ where $\mathcal{R}_{k,l} = \{k, l : k - r/2 \leq k \leq k + r/2 - 1, l - r/2 \leq l \leq l + r/2 - 1\}$ and for the component h that matches the sample in the

neighbourhood, $z_{n,k,l,h} = 1$, otherwise, the value is zero. We can model the samples as

$$p(\boldsymbol{y}_{n,i,j}|\theta) = \sum_{\mathcal{R}_{k,l}} \sum_{h} \omega_{k,l,h} \mathcal{N}(\boldsymbol{y}_{n,i,j}|\mu_{k,l,h}, \Sigma_{k,l,h}) \tag{3}$$

4.1. Expectation Step (E-Step)

The likelihood of the complete data including the latent variables, i.e. (Y, Z) can be expressed as the probability of the complete data, given the associated parameter set θ [21].

$$L(\theta; Y, Z) = p(Y, Z|\theta) \tag{4}$$

Now, the EM algorithm helps to find the expected value of the logarithm of this likelihood function. Since the values of the latent variable Z are unknown, we compute this expectation under the posterior probability of the latent variable Z given the observed data Y and the current parameter estimates θ^{old}, which is denoted as $Q(\theta, \theta^{old})$. The argument θ in the $Q(.,.)$ function is the new set of estimated parameter values that we aim to optimise in order to maximise this function and θ^{old} is the set of current parameter estimates with which the expectation is evaluated.

$$\begin{aligned} Q(\theta, \theta^{old}) &= E_{Z|Y}(\log p(Y, Z|\theta)) \\ &= \sum_{Z} p(Z|Y, \theta^{old}) \log p(Y, Z|\theta) \end{aligned} \tag{5}$$

In equation (5), the term $p(Z|Y, \theta^{old})$ is the posterior probability distribution of the latent variables given the data and the parameter estimates and here, Z below the summation indicates that the expectation is calculated over all possible states of the latent variable. This posterior probability

can be calculated for each mixture $z_{k,l,h}$ belonging to the latent variable Z_n according to Bayes' rule since we can assume that the prior distribution is given by the weight estimates of the mixtures and the likelihood is given by the probability of the observed data with respect to the parameter estimates marginalised over the particular mixture $z_{k,l,h}$.

Now, while the expectation of the term $z_{k,l,h}$ under this distribution is calculated over all possible states of the latent variable, it will be equal to the probability when the mixture is selected, i.e.$p(z_{k,l,h} = 1|\boldsymbol{y}_{i,j}, \theta^{old})$, since the value of the other terms when the mixture is not selected will be equal to zero.

Hence, for standard mixture modelling, this is given by,

$$E_{p(z_{k,l,h}|y_{i,j},\theta^{old})}(z_{k,l,h}) = \frac{\omega_{k,l,h} * \mathcal{N}(y_{i,j}|\mu_{k,l,h}, \Sigma_{k,l,h})}{\sum_{h'=1}^{H} \omega_{k,l,h'} * \mathcal{N}(y_{i,j}|\mu_{k,l,h'}, \Sigma_{k,l,h'})} \tag{6}$$

Note, however, that in the case of the standard mixture modelling, $(i, j) \equiv (k, l)$, i.e. the mixtures are located in the same location as the data samples.

In the case of region-based modelling, the probability of the samples is governed by equation (3). Therefore, a second summation term would be required in the denominator. Also, since the mixture component $z_{k,l,h}$ contributes to the samples in the neighbourhood $\mathcal{R}_{k,l}$, the total contribution is computed as the summation of all contributions in the neighbourhood, which is represented by the additional summation term over the whole expression.

$$E_{p(z_{k,l,h}|\boldsymbol{y}_{i,j},\theta^{old})}(z_{k,l,h}) = \sum_{\mathcal{R}_{k,l}} \frac{\omega_{k,l,h} * \mathcal{N}(\boldsymbol{y}_{i,j}|\mu_{k,l,h}, \Sigma_{k,l,h})}{\sum_{\mathcal{R}_{k,l}} \sum_{h'=1}^{H} \omega_{k,l,h'} * \mathcal{N}(\boldsymbol{y}_{i,j}|\mu_{k,l,h'}, \Sigma_{k,l,h'})}$$
$$= \sum_{\mathcal{R}_{k,l}} \gamma_{i,j}(z_{k,l,h}) \tag{7}$$

Here, γ denotes the contribution of the mixture component $z_{k,l,h}$ to the sample $\boldsymbol{y}_{i,j}$. The subscript for γ here that indicates that the expectation of the mixtures based on each of the data sample in its neighbourhood is maintained separately. Outside of the cluster's neighbourhood, the data does not influence the cluster, therefore, the expectation values are assumed to be known and zero.

Now, the complete likelihood term $p(\boldsymbol{y}, \boldsymbol{z}|\theta)$ can be rewritten as

$$p(\boldsymbol{y}, \boldsymbol{z}|\theta) = p(\boldsymbol{z}|\theta)\, p(\boldsymbol{y}|\boldsymbol{z}, \theta) \tag{8}$$

The prior distribution of the hidden variables in the region is given by

$$p(\boldsymbol{z}_{k,l}) = \prod_{\mathcal{R}_{k,l}} \prod_{h} (\omega_{k,l,h})^{z_{k,l,h}} \tag{9}$$

Here, $z_{k,l,h}$ assumes a value of either one or zero depending on whether the mixture is chosen or not.

Also, the probability of the pixel given the data cluster $\{k, l, h\}$ is given by,

$$p(\boldsymbol{y}_{i,j}|\boldsymbol{z}_{k,l}) = \prod_{\mathcal{R}_{k,l}} \prod_{h} (\mathcal{N}(\boldsymbol{y}_{i,j}|\mu_{k,l,h}, \Sigma_{k,l,h}))^{z_{k,l,h}} \tag{10}$$

This probability will hold within the neighbourhood because the weights are normalised over the region rather than at the particular location.

Now, substituting equations (9) and (10) in equation (8), we get

$$p(\boldsymbol{y}_{i,j}, \boldsymbol{z}_{k,l}) = \prod_{\mathcal{R}_{k,l}} \prod_{h} (\omega_{k,l,h} \mathcal{N}(\boldsymbol{y}_{i,j} | \mu_{k,l,h}, \Sigma_{k,l,h}))^{z_{k,l,h}} \tag{11}$$

The complete likelihood function in equation (5) can then be written as

$$p(Y, Z) = \prod_{n} p(\boldsymbol{y}_n, \boldsymbol{z}_n) \tag{12}$$

Plugging (11) into (12) leads to

$$p(Y, Z) = \prod_{n} \prod_{\mathcal{R}_{k,l}} \prod_{h} (\omega_{k,l,h} \mathcal{N}(\boldsymbol{y}_{n,i,j} | \mu_{k,l,h}, \Sigma_{k,l,h}))^{z_{n,k,l,h}} \tag{13}$$

Taking logarithm on both sides of equation (13), the log likelihood function is given by

$$\log p(Y, Z) = \sum_{n} \sum_{\mathcal{R}_{k,l}} \sum_{h} z_{n,k,l,h} \\ * \{\log \omega_{k,l,h} + \log \mathcal{N}(\boldsymbol{y}_{n,i,j} | \mu_{k,l,h}, \Sigma_{k,l,h})\} \tag{14}$$

Using equations (7) and (14) in the Q function gives

$$Q(\theta, \theta^{old}) = \sum_{n} \sum_{\mathcal{R}_{k,l}} \sum_{h} \gamma_{i,j}(z_{n,k,l,h}) \\ * \{\log \omega_{k,l,h} + \log \mathcal{N}(\boldsymbol{y}_{n,i,j} | \mu_{k,l,h}, \Sigma_{k,l,h})\} \tag{15}$$

In the case of traditional MoG, the above equation can be derived similarly as

$$Q(\theta, \theta^{old}) = \sum_{n} \sum_{h} \gamma(z_{n,h}) * \{\log \omega_h + \log \mathcal{N}(y_n | \mu_h, \Sigma_h)\} \tag{16}$$

Here, we have omitted the pixel and mixture parameter indices to show that they both belong to the same location. It can be seen that the first summation becomes degenerate here since the region size r is one.

In case of hard EM implementations, the value of $\gamma_{i,j}(z_{k,l,h})$ (from equation (7)) becomes one only for the mixture chosen and zero for the remaining mixtures.

Equation (15) is the Q function to be maximised with respect to each of the parameters in the parameter set θ in the M-step of the algorithm.

4.2. Maximisation Step (M-Step)

In equation (15) , the distribution $\mathcal{N}$ can be expanded as

$$\mathcal{N}(\boldsymbol{y}_{i,j}|\boldsymbol{\mu}_{k,l,h}, \Sigma_{k,l,h}) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma_{k,l,h}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{y}_{i,j}-\boldsymbol{\mu}_{k,l,h})^T\Sigma^{-1}_{k,l,h}(\boldsymbol{y}_{i,j}-\boldsymbol{\mu}_{k,l,h})} \quad (17)$$

Taking logarithms on both sides and substituting in equation (15), we get

$$\begin{aligned} Q(\theta, \theta^{old}) = \sum_n \sum_{\mathcal{R}_{k,l}} \sum_h \gamma_{i,j}(z_{n,k,l,h}) \\ * \{\log \omega_{k,l,h} - \frac{D}{2}\log(2\pi) + \frac{1}{2}\log\left|\Sigma^{-1}_{k,l,h}\right| \\ - \frac{1}{2}(\boldsymbol{y}_{i,j} - \boldsymbol{\mu}_{k,l,h})^T\Sigma^{-1}_{k,l,h}(\boldsymbol{y}_{i,j} - \boldsymbol{\mu}_{k,l,h})\} \quad (18) \end{aligned}$$

Now, in order to obtain the update equations of the different parameters, this equation (18) is differentiated partially with respect to the corresponding parameter and the equation set to zero.

4.3. Batch Mode Update Parameters

4.3.1. Mean of the distribution ($\boldsymbol{\mu}$)

Taking derivatives of equation (18) with respect to the mean ($\boldsymbol{\mu}$) and setting the derivative to zero gives

$$\boldsymbol{\mu}_{k,l,h} = \frac{1}{\sum_n \sum_{\mathcal{R}_{k,l}} \gamma_{i,j}(z_{n,k,l,h})} \sum_n \sum_{\mathcal{R}_{k,l}} \gamma_{i,j}(z_{n,k,l,h})\, \boldsymbol{y}_{n,i,j} \tag{19}$$

Now, the number of samples in a given cluster is given by

$$N_{k,l,h} = \sum_n \sum_{\mathcal{R}_{k,l}} \gamma_{i,j}(z_{n,k,l,h}) \tag{20}$$

Substituting equation (20) into equation (19)

$$\boldsymbol{\mu}_{k,l,h} = \frac{1}{N_{k,l,h}} \sum_n \sum_{\mathcal{R}_{k,l}} \gamma_{i,j}(z_{n,k,l,h})\, \boldsymbol{y}_{n,i,j} \tag{21}$$

4.3.2. Variance of the distribution (Σ)

Taking derivatives of equation (18) with respect to the variance Σ and setting the derivative to zero, we get

$$\Sigma_{k,l,h} = \frac{1}{N_{k,l,h}} \sum_n \sum_{\mathcal{R}_{k,l}} \gamma_{i,j}(z_{n,k,l,h})(\boldsymbol{y}_{n,i,j} - \boldsymbol{\mu}_{k,l,h})^T(\boldsymbol{y}_{n,i,j} - \boldsymbol{\mu}_{k,l,h}) \tag{22}$$

4.3.3. Weight of the distribution (ω)

For deriving the weight of the distribution, there is an additional constraint that the weights are normalised over the region. This implies that $\sum_{\mathcal{R}_{k,l}} \sum_h \omega_{k,l,h} = 1$. Hence, in order to perform the differentiation, we construct a Lagrangian multiplier with this constraint.

$$L = Q + \lambda(1 - \sum_{\mathcal{R}_{k,l}} \sum_h \omega_{k,l,h}) \tag{23}$$

Now, differentiating this equation partially with respect to the weight ω_{qk},

$$\frac{\partial L}{\partial \omega_{k,l,h}} = \sum_{n} \gamma_{i,j}(z_{n,k,l,h}) \frac{1}{\omega_{k,l,h}} - \lambda \tag{24}$$

Setting this equation to zero, we get

$$\omega_{k,l,h}\lambda = \sum_{n} \gamma_{i,j}(z_{n,k,l,h}) \tag{25}$$

Summing this equation over all the Gaussians in the region,

$$\sum_{\mathcal{R}_{k,l}} \sum_{h} \omega_{k,l,h}\lambda = \sum_{n} \sum_{\mathcal{R}_{k,l}} \sum_{h} \gamma_{i,j}(z_{n,k,l,h}) \tag{26}$$

In the left side of equation (26), we know the sum of all the mixture weights in the neighbourhood is equal to one, i.e. $\sum_{\mathcal{R}_{k,l}} \sum_{h} \omega_{k,l,h} = 1$. Also, the term on the right side of equation (26) is equal to the total number of samples in the region over the entire data. It follows from this that

$$\lambda = r^2 \, N \tag{27}$$

Substituting for λ in equation (25) gives the mixture weight as

$$\omega_{k,l,h} = \frac{N_{k,l,h}}{r^2 \, N} \tag{28}$$

The equations (21), (22) and (28) are the batch mode update equations for the mean, variance and mixture weights respectively.

The mean and variance equations are similar to those of the traditional approach; however the weights are normalised over the neighbourhood. This shows that the equations degenerate to the original EM equations when the neighbourhood size is one.

4.4. Online Mode Update Parameters

For the online mixture parameter updates, a stochastic gradient descent method is applied [22] which is of the form

$$\theta^t = \theta^{t-1} + \alpha_\theta^t(\nabla_\theta Q) \tag{29}$$

For online mode updates, only one observation is used at any instant along with the previously modelled samples. Therefore equation (15) is correspondingly modified as

$$Q(\theta, \theta^{old}) = \sum_{\mathcal{R}_{k,l}} \sum_{h} \gamma(z_{k,l,h}) \{\log \omega_{k,l,h} + \log \mathcal{N}(\boldsymbol{y}_{t,i,j} | \boldsymbol{\mu}_{k,l,h}, \Sigma_{k,l,h})\} \tag{30}$$

Now, for the mean μ, equation (29) is updated as

$$\boldsymbol{\mu}_{k,l,h}^t = (1-\rho)\boldsymbol{\mu}_{k,l,h}^{t-1} + \rho(\boldsymbol{y}_{i,j}^t) \tag{31}$$

For the variance Σ, equation (29) becomes

$$\Sigma_{k,l,h}^t = (1-\rho)\Sigma_{k,l,h}^{t-1} + \rho(\boldsymbol{y}_{i,j}^t - \boldsymbol{\mu}_{k,l,h}^{t-1})^2 \tag{32}$$

In the case of the mixture weights ω, equation (29) becomes

$$\omega_{k,l,h}^t = (1-\alpha)\omega_{k,l,h}^{t-1} + \alpha \tag{33}$$

The mixture weights should be normalised over the entire region $\mathcal{R}_{k,l}$ to make sure they sum up to one. In equations (31), (32) and (33), ρ is the learning rate of the distribution, α is the learning rate of the mixture weights and t denotes the time instant.

When comparing these equations with the online update equations of the traditional MoG method, it can be seen that the mean and variance equations are the same; however, the weight equation is slightly different, as in the

case of the batch mode equations, where they are normalised over the whole region rather than at the particular pixel location. These are also similar to the pure hard update equations of [18], except that in their equations the exponential decay is applied to the time-weighted sample variables that are used to calculate the parameters, whereas in these equations the exponential decay is directly applied to the parameters. This helps in reducing the complexity of the algorithm because the additional variables are not calculated during each instant in this method.

5. Region-based Mixture of Gaussians for foreground segmentation

In this section, we describe the RMoG algorithm for foreground detection in video sequences. The model is updated by using the online update equations given in the previous section. Here, the pixels/blocks are not independent of each other, and correspondingly the updated mixtures represent the scene distribution in a neighbourhood region. The background model formed by this approach is not just the dominant clusters of the individual pixel or block under consideration; rather it is a collection of all the background distributions in the region defined by the neighbourhood. Hence, when a new observation is encountered, if it falls under any of these distributions it is classified as a background pixel/block as illustrated in Fig. 2. Even though foreground pixels/blocks may also have similar distributions in a region, they are not classified as background because of their low weight and high variance.

As discussed previously, this algorithm is not restricted to a single pixel in a neighbourhood, rather it can handle feature vectors that consist of blocks of

size $f \times f$, such as 2×2, 4×4 etc. The mean vector for each mixture of a block is equal to the size of the block feature vector whereas a scalar variance and weight is maintained for the mixtures. The variance is scaled appropriately depending on the size of the feature block. The reference block can be chosen arbitrarily within the region, however, in our case, we chose the central block of the region or one close to the centre depending on whether the region size is odd or even.

These block based features have several advantages over single pixel features in that they are more robust to noise due to camera jitter and also due to illumination changes [23]. While the output of the algorithm using block based features may be coarser compared to using pixel level features, the resultant definition is sufficient for other applications such as human detection and tracking. In a broader sense, this overall framework could be looked upon as a bridge between a pixel level algorithm to a block level algorithm. The choice of the region size and the feature size depends on the requirements of the application.

Algorithm: Region-based Mixture of Gaussians (RMoG)

1. Consider a series of T images $Y = \{Y_1, Y_2, ..., Y_t, ..., Y_T\}$ where t is the index of the image at the current time instant. The image Y_t (of size $I \times J$ feature blocks) at location (i, j) can be denoted by $Y_t = \{\boldsymbol{y}_{t,i,j} : i = 1 : I, j = 1 : J\}$ where $\boldsymbol{y}_{t,i,j}$ are the different feature vectors. The model is given by the parameters $\{\theta_{t,i,j,h} : \boldsymbol{\mu}_{t,i,j,h}, \Sigma_{t,i,j,h}, \omega_{t,i,j,h}\}$ where h=1:H is the index of the mixture and H is the total number of mixture components at each location.

2. Initialise the model parameters as follows: $\boldsymbol{\mu}_{t,i,j,1} = \boldsymbol{y}_{1,i,j}$, $\Sigma_{t,i,j,1}$ propor-

tional to the size of the feature vector, and $\omega_{1,i,j,1} = 1$ and $\omega_{1,i,j,2:H} = 0$.

3. For every subsequent image Y_t, calculate the most likely Gaussian mixture $\theta_{t,k,l,h}$ where $(k,l) \in \mathcal{R}_{i,j}$ for the reference feature block $\boldsymbol{y}_{i,j}$ (over its entire neighbourhood). This can be calculated by using Euclidean distance.

4. Compare the distance of the most likely Gaussian mixture component with a threshold equal to a constant D times the standard deviation of the mixture component. This indicates whether the pixel sample matches the mixture component or falls outside the mixture model.

5. If a match is found, update the mixture parameters of the above Gaussian $\theta_{t,k,l,h}$ by using the equations (31)-(33). The weights are normalised such that the weights for each mixture component at position (i,j) sum to one, i.e., $\sum_h \omega_{t,i,j,h} = 1$.

6. If no match is found, create a new Gaussian mixture component, if there already aren't H mixtures at reference block location (i,j), or else replace the Gaussian mixture component with the lowest weight (at the current block location (i,j)) by initialising a new mixture component.

7. The background model is built by ranking the components according to $\omega_{t,i,j,h}/\Sigma_{t,i,j,h}$ and selecting the top n ranked components such that their added ranking coefficients are greater than some threshold. If the observation falls within this model, it is classified as a background pixel; otherwise it is classified as foreground.

5.1. Complexity Analysis of Region-based Mixture of Gaussians (RMoG)

In this section, the computational complexity of the RMoG algorithm is described for different region and feature size combinations, including the standard MoG algorithm. In particular, we study the asymptotic behaviour of the algorithm for each image in terms of the "big O" notation. Before introducing the complexity, the different notations used in the analysis are reiterated here. The image Y_t is of size $N = I \times J$ pixels. H is the total number of mixture components at each location. The region size is $r \times r$ and the feature size is $f \times f$. It has to be noted that f is always a factor of r and cannot be greater than r. If f is greater than 1, then each feature vector is an $f \times f$ block. This implies that the total number of blocks to be processed in each image are $\frac{N}{f^2}$. It can be seen that the number of blocks to be processed per image decreases with increasing f value.

The RMoG algorithm can be divided into two main steps for the sake of complexity analysis. The first step is matching the current sample with the most likely Gaussian mixture in its neighbourhood. The worst case complexity of this step is $\mathcal{O}\left(H\frac{r^2}{f^2}\right)$. The second step is sorting the background mixtures according to their likelihood and the worst case complexity of the sorting algorithm is given by $\mathcal{O}\left(H \log H\right)$. Therefore the complexity of the algorithm for a single feature block is $\mathcal{O}\left(H\left(\frac{r^2}{f^2} + \log H\right)\right)$. Since the total number of blocks to be processed in an image is given by $\frac{N}{f^2}$, the total complexity of the algorithm for an image is given by $\mathcal{O}\left(\frac{NH}{f^2}\left(\frac{r^2}{f^2} + \log H\right)\right)$. The computational complexities for the different region size and feature size combinations can be obtained by substituting the values for r and f in this expression.

Feature size and Region size combinations	Complexity
r=1, 1×1 feature block	$\mathcal{O}\left(NH\left(1+\log H\right)\right)$
r>1, 1×1 feature block	$\mathcal{O}\left(NH\left(r^2+\log H\right)\right)$
r>1 and r=f, $f\times f$ feature block	$\mathcal{O}\left(\frac{NH}{f^2}\left(1+\log H\right)\right)$
r>1 and $r\neq f$, $f\times f$ feature block	$\mathcal{O}\left(\frac{NH}{f^2}\left(\frac{r^2}{f^2}+\log H\right)\right)$

Table 1: Computational Complexity of the RMoG algorithm

Table 1 lists the computational complexity for all possible combinations of region sizes and feature sizes in the RMoG algorithm. The first row in this table corresponds to the standard MoG algorithm when r=1 and f=1. The second row corresponds to the RMoG algorithm with a single pixel as the feature vector, i.e. f=1 within a given region $\mathcal{R}$. It can be seen that the complexity of the algorithm increases with region size, as the matching process takes place within an entire region, making it slower than the traditional MoG algorithm. However, if the feature vector is increased to the same size of the region, i.e. r=f (Third row in Table 1), the RMoG algorithm becomes much faster than the MoG algorithm, albeit at the expense of poorer foreground definition. The most general case is shown in the fourth row of Table 1 where it can be easily deduced that for a given value of r, increasing the value of f decreases the computational complexity of the RMoG algorithm.

Sequence Name	Sample Frame	Image Size	Number of Frames
Bottle [24]		240×320	110
Beach [25]		128×160	150
Waving Trees [26]		120×160	120
Bus		240×320	200
Boats [27]		240×320	200
Canoe [27]		240×320	300
Fall [27]		240×320	200
Fountain1 [27]		240×320	150
Fountain2 [27]		240×320	150
Overpass [27]		240×320	300

Table 2: Details of the different video sequences

6. Experiments and Results

To evaluate the algorithm we tested it using ten different sequences containing dynamic backgrounds, most of which have been used extensively by the video analytics research community. The details of these ten sequences are given in Table 2. The first two of these sequences, Bottle sequence [24] and Beach sequence [25], have dynamic background in the form of rippling water surfaces. The waving trees sequence [26] is self-explanatory. Then, we introduce here to the foreground detection community, a sequence taken from a CCTV camera onboard a moving bus. This sequence contains moving background regions in the window areas of the bus. In addition, there are also dramatic illumination changes occurring inside the bus due to fast moving shadows caused by the motion of the bus relative to the sun. The sequence is very complex, extremely challenging and unlike others used before. We make it and its ground truth available to others. The remaining six sequences are from the Dynamic Background category of the ChangeDetection benchmark dataset [27]. These sequences contain different scenarios where there is strong dynamic motion in the background such as boats moving through water (Boats sequence and Canoe sequence), cars passing behind fountains (Fountain1 and Fountain2 sequences) and cars and pedestrians passing through a scene with trees shaking in the wind (Fall sequence and Overpass sequence).

In each of the experiments, we varied the detection threshold and calculated for each image the probability of detection (True Positive Rate) given by $TPR = TP/(TP+FN)$ and probability of false alarm (False Positive Rate) given by $FPR = FP/(FP+TN)$ where *TP, FP, TN and FN* are the number

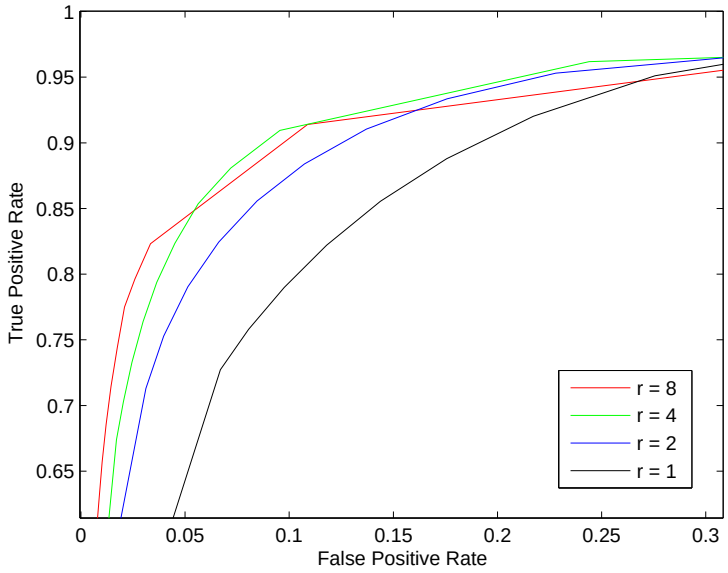


Figure 3: Combined ROC curve for the ten video sequences for different region sizes with a single pixel as the feature. (Black corresponds to Original MoG) (Best viewed in color)

of True Positives, False Positives, True Negatives and False Negatives respectively. The Receiver Operating Characteristic (ROC) Curves were then obtained by plotting the TPR versus the FPR. These ROC curves help illustrate the performance of binary classifier systems, in our case the Foreground-Background classifier.

6.1. Investigation into the effect of region size variation

In the first experiment, we varied r=8, 4, 2, and 1, while keeping the feature fixed to a single pixel, i.e. f=1. The ROC curve obtained by combining the results of all the sequences, Fig. 3, generally shows that as the region size is increased foreground detection is improved. In particular, the algorithm performs better in the region of low false positives. This is further

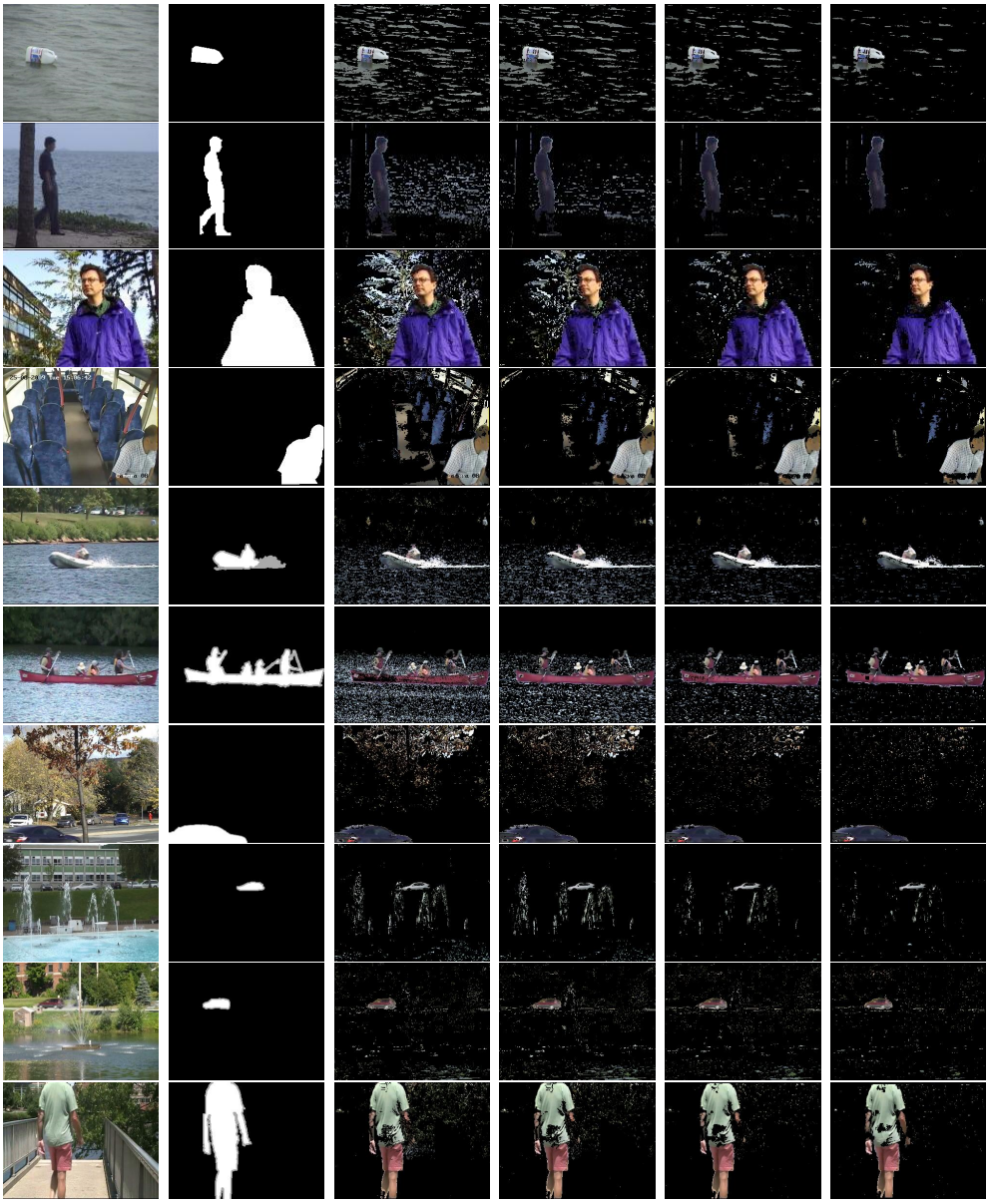


Figure 4: Qualitative results for the different video sequences. First Column: Input frames; Second Column: Ground Truth; Third Column: Original MoG (r=1, 1×1 feature block); Fourth Column: RMoG (r=2, 1×1 feature block); Fifth Column: RMoG (r=4, 1×1 feature block); Sixth Column: RMoG (r=8, 1×1 feature block)

illustrated in Fig. 4 which shows the output obtained for a single image from each sequence with different region sizes. It is clear that as the region size increases the number of false alarms arising from the dynamic background motion reduces. In the bottle sequence, Fig. 4 (top row), the water ripples are subtracted almost entirely with r=8, while the bottle is still detected as foreground. In the beach sequence, Fig. 4 (2nd row), similar results are obtained with r=4. This can be attributed to the fact that the frames in the bottle sequence are almost double the size of the beach sequence frames. Therefore, the dynamic motion in the bottle sequence spans across a larger region than the beach sequence. In the waving trees sequence, Fig. 4 (3rd row), the algorithm is able to progressively subtract more of the complex background with increasing region size. However, an interesting point to note here is that there is a slight decrease in the number of true positives, especially with a larger region size. This can be explained from the fact that the colour of the person's shirt matches the colour of the leaves. As the region size increases, the likelihood of them containing both sets of pixels increases, hence shirt pixels are associated with the models of the leaf pixels. This can also be noted in the bus sequence, Fig. 4 (4th row), where the hair colour of the person matches the background in that region, thus resulting in false negatives. However, this increase in false negatives is not significant compared to the decrease in false positives. This can be confirmed from Table 3 where the average value of the percentage of false negatives and false positives are shown for each sequence. Here, for the waving trees and bus sequences, the percentage increase of false negatives is much lower compared to the percentage decrease of false positives as the region size is changed

from r=1 to r=8. Another advantage of this method is that it is quite robust to illumination changes and this can be seen particularly in the aisle of the moving bus sequence, Fig. 4 (4th row). The original MoG approach is known to not handle significant changes in illumination whereas by using the spatial correspondence between the pixels, the RMoG model is quite capable of handling changes in the scene lighting. In the Boats sequence, Fig. 4 (5th row), and the Canoe sequence, Fig. 4 (6th row), the dynamic background from the waves is again handled well with an increase in the region size. This is similar to the behaviour previously seen in the Bottle and Beach sequences. The difference in performance for different region sizes in Fountain1, Fig. 4 (8th row), and Fountain2, Fig. 4 (9th row), are not very pronounced. This is because of the low number of false positives in the sequence. However, on closer inspection, it can be seen that increasing the region size still helps decrease the number of false alarms due to the dynamic background. The performance of the algorithm in the Fall sequence, Fig. 4 (6th row), and the Overpass sequence, Fig. 4 (10th row) are seen to be similar to the performance in the Waving Trees sequence where there is a similar type of dynamic background, i.e. trees swaying in the wind.

6.2. Investigation of effect of feature size variation

We then experimented by keeping the region size constant while varying the size of the feature vector. The RMoG algorithm, with a region size of 8×8, was tested against features of a single pixel, a 2×2 block, a 4×4 block and an 8×8 block of pixels. The performance of this experiment on the ten video sequences can be seen in the combined ROC curve in Fig. 5. Here, there is no apparent trend in the influence of the feature size on the

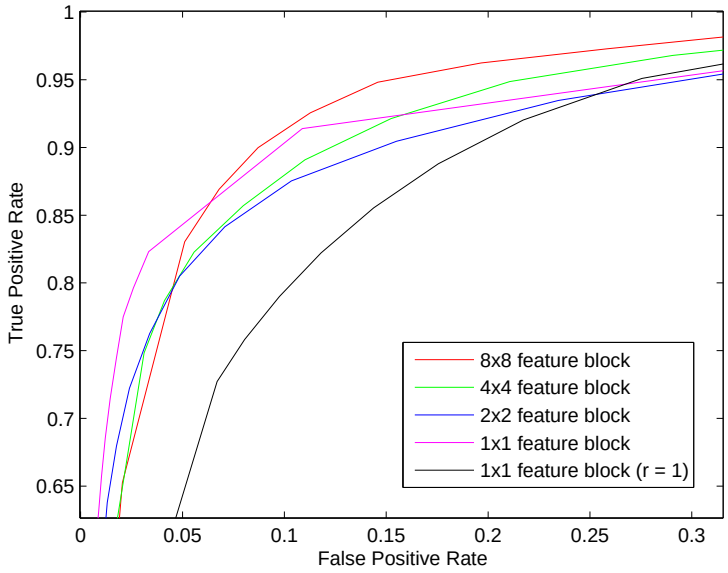


Figure 5: Combined ROC curve for the ten video sequences for different feature sizes in an 8×8 region. (Black corresponds to Original MoG) (Best viewed in colour)

algorithm, except that all cases with increased region size work better than the original MoG. However, this does not tell the whole story. Figure 6 shows an example frame from the optimum result for each sequence. Obviously, for larger feature sizes there will be larger blocks of false positives, i.e., a feature size of 2×2 gives rise to a false positive block of 2×2 and so on. On closer inspection of the Boats sequence images, Fig. 6 (5th row), it can be seen that for the smaller feature sizes (2nd and 3rd columns), the false positives are obviously smaller, greater in number and widely spread, whereas increasing the feature size (7th column), obviously results in larger false positives, which are fewer and more sparsely distributed. Thus, although the total number of pixels that are classified as false positives is roughly the

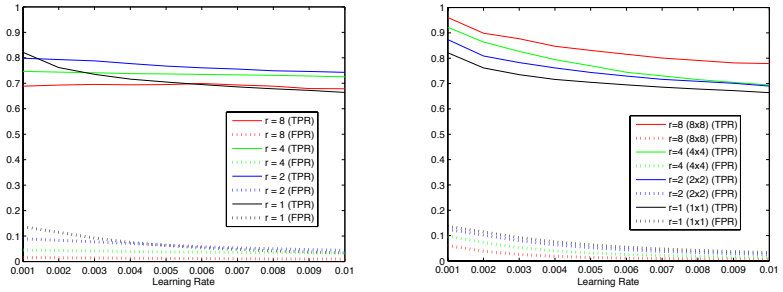
Figure 6: Qualitative results for the different video sequences. First Column: Input frames; Second Column: Ground Truth; Third Column: Original MoG (r=1, 1×1 feature block); Fourth Column: RMoG (r=8, 1×1 feature block); Fifth Column: RMoG (r=8, 2×2 feature block); Sixth Column: RMoG (r=8, 4×4 feature block); Seventh Column: RMoG (r=8, 8×8 feature block)

same for both cases, their characteristic patterns of background clutter in the segmented image are quite different. Furthermore, the blockier nature of larger feature sizes means that a region deemed as a true positive on the boundary between the foreground and background, may give rise to increased false positives. This is illustrated in the Beach sequence, Fig. 6 (2nd row, 7th column), where the foreground object definition is quite blocky giving rise to false positives. However, note that there are no false positives due to the dynamic background. Conversely, for a smaller feature size, column 4, the foreground object definition is better, with false positives in this case being due to the dynamic background. Thus although the cause of the false positives is different in both cases, the overall number of pixels classified as false positive is roughly the same. These could explain why the ROC curve in Fig. 5 does not show any trend with an increase in feature size.

We also investigated other region and feature size combinations, such as 2×2, 2×4 and 4×4, and drew similar conclusions in relation to the effect of the feature size.

6.3. Investigation of the effect of learning rate

There has been extensive research on the learning rate schedule for MoG (section 3.4 of [6] focuses on the different approaches to adapt the learning rate in MoG). While all the experiments in sections 6.1 and 6.2 use the same learning rate in order to illustrate the effect of the region and feature sizes, we also investigated the sensitivity of our approach to learning rate. To begin with, we first investigated how learning rate affected the performance of our approach for different regions sizes with a single pixel as the feature vector. Figure 7(a) shows the plot of true positive rates and false positive



(a) (b)

Figure 7: Plot of True Positive Rates and False Positive Rates for different values of learning rates when (a) the feature size is equal to a single pixel and (b) the region size is equal to the feature size

rates for learning rates between 0.001 and 0.01. Analysis shows that for a region size of 1×1, both the true and false positive rates decrease as the learning rate increases. This is because, by using a large learning rate, the objects in the foreground are taken to the background almost immediately. However, as the region size increases the effect of learning rate diminishes until for r=8 it has no discernible effect on performance. A possible reason for this is, with a feature size of one pixel, as the region size increases the number of mixture components in the region-based model increases, so that the number of updates for each individual component decreases, thereby reducing the effect of the learning rate. To confirm this we carried out a second experiment, similar to the first, but in which the region size and feature size were made equal, Fig. 7(b). As expected, it can be seen that as the learning rate increases, the number of true and false positives decrease in all cases. This is because, for different region sizes, the number of components in the mixture model is the same when the feature and region sizes are equal.

6.4. Comparison with other foreground detection algorithms

We compared our results with the results from other well-known background subtraction approaches such as Eigenbackground [28], ViBe [29], MGM-UM [30] and PBAS [31]. The Eigenbackground provides a robust model of the background by using Eigen decomposition of the images and reducing the resultant dimensionality by using Principal Component Analysis. ViBe is currently one of the best background subtraction algorithms in the literature. It is an adaptive model that is maintained with a pool of samples and the model is updated by randomly replacing a sample from the pool with the observed sample. This is different from most modelling approaches where the model is updated in a way that the least contribution is from the oldest sample. This method is quite effective in handling both illumination and dynamic changes in the background. MGM-UM is a fuzzy variation of the MoG approach with an uncertain mean vector controlled by a fuzzy parameter. This approach aims to estimate the parameters better than the original approach even with noisy real-world data. 'Pixel-Based Adaptive Segmenter' or PBAS is a non-parametric state of the art method that uses a random pool of samples from the history to maintain the background model and two controllers with feedback to update the decision threshold and the learning parameter.

In Fig. 8, the input images and the ground truths are shown in the first two columns. The standard MoG results are shown in the third column followed by the outputs of Eigenbackground, ViBe, MGM-UM and PBAS in the fourth, fifth, sixth and seventh columns respectively. The software for ViBe was obtained from the authors' webpage [32] while the software

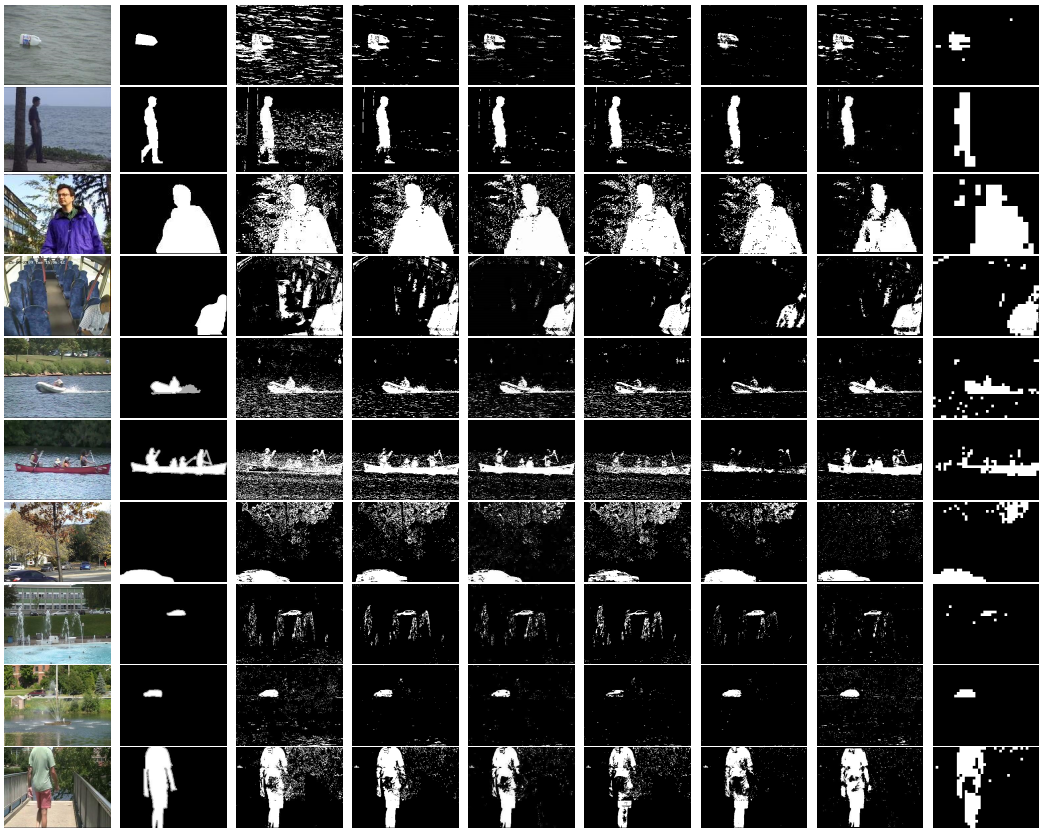


Figure 8: Comparison of outputs from different foreground detection algorithms. First Column: Input frames; Second Column: Ground Truth; Third Column: Original MoG (r=1, 1×1 feature block); Fourth Column: Eigenbackground [28]; Fifth Column: ViBe [29]; Sixth Column: MGM-UM [30]; Seventh Column: PBAS [31]; Eighth Column: RMoG (r=8, 1×1 feature block); Ninth Column: RMoG (r=8, 8×8 feature block)

Algorithm	Complexity	Errors	Sequence									
			Bottle	Beach	Trees	Bus	Boats	Canoe	Fall	Fountain1	Fountain2	Overpass
MoG (r=1, f=1)	$\mathcal{O}(NH(1+\log H))$	FN	0.48	0.67	0.96	0.42	1.11	3.72	0.63	0.41	0.12	4.58
		FP	17	9.23	16.96	14.75	10.02	13.69	9.83	3.41	3.77	5.01
RMoG (r=8, f=1)	$\mathcal{O}(NH(64+\log H))$	FN	0.77	2.21	5.18	1.98	1.79	4.14	1.21	0.55	0.14	4.31
		FP	3.51	0.42	1.12	2.36	1.73	1.21	1.98	1.01	2.12	0.64
RMoG (r=8, f=8)	$\mathcal{O}\left(\frac{NH}{64}(1+\log H)\right)$	FN	0.42	0.3	0.83	0.7	0.52	6.42	0.35	0.38	0.04	4.96
		FP	1.52	3.12	6.03	3.01	3.73	1.45	5.49	0.44	0.35	2.17
Eigenbackground	$\mathcal{O}(N^2M+N^3)$	FN	0.51	1.32	0.32	0.61	1.48	3.55	1.06	0.55	0.33	5.47
		FP	2.73	1.08	10.32	5.69	3.61	6.55	9.84	2.33	0.56	2.03
ViBe	$\mathcal{O}(NB)$	FN	0.77	1.63	1.15	1.56	1.88	4.38	0.55	0.52	0.28	6.15
		FP	2.81	0.82	7.73	2.2	2.94	2.63	4.91	1.7	0.4	0.95
MGM-UM	$\mathcal{O}(NH(1+\log H))$	FN	0.73	1.63	0.9	1.87	2.52	7.33	2.34	0.75	0.8	9.37
		FP	5.59	2.06	11.6	2.06	5.67	5.16	11.93	3.03	0.42	2.23
PBAS	$\mathcal{O}(NB)$	FN	1.2	2.42	1.68	5.84	3.09	10.55	0.61	0.36	0.25	7.44
		FP	0.88	1.56	8.04	0.66	0.88	0.49	5.67	1.03	0.21	0.58

Table 3: Computational Complexity and Quantitative Results for the different algorithms

for Eigenbackground, MGM-UM and PBAS were obtained from Andrews Sobral's background subtraction library [33]. The last two columns show the results for the RMoG algorithm with r=8 and block feature sizes of 1×1 and 8×8 respectively. Table 3 gives the performance in terms of the percentage of False Positives and False Negatives as well as the computational complexities of all the algorithms compared.

For the Bottle sequence, top row, the RMoG with a feature size of 8×8 removes almost all of the background, apart from the bottle reflection on the water surface. The foreground object is well defined internally, however, there is some loss in shape detail at its boundary. PBAS also removes the dynamic background, but there are a large number of false negatives as well. Comparing it with the statistics in Table 3, RMoG with 8×8 feature block has the lowest percentage of False Negatives while PBAS has the lowest per-

centage of false positives closely followed by RMoG with 8×8 feature block. The next best results are obtained for Eigenbackground and ViBe. In these cases there is more background present, along with the bottle reflection. The internal definition is poorer, as there are holes due to the bottle packaging being incorrectly classified as background. Next best, with slightly more background clutter, is RMoG with a feature size of 1×1. After that comes MGM-UM with more background clutter. The worst result is obtained with the original MoG. Similar trends can be observed for the Beach sequence in the 2nd row, where RMoG with a feature size of 8×8 is able to subtract the dynamic background region. The 3% false positives attributed to it in Table 3 is mainly due to the false positives arising due to the blocky output around the edges of the person. This phenomenon was discussed at length in section 6.2. In this sequence, RMoG with a feature size of 1×1 has less background clutter than either Eigenbackground or ViBe and is similar to PBAS. For the Waving Trees sequence, RMoG with feature sizes of 1×1 and 8×8 both remove most of the background. None of the other techniques, columns 3-7, has much success with removing the background, although their foreground definition is better. This is also observed in Table 3 where the percentage of False Positives is the lowest for the two RMoG cases. For the Bus sequence, RMoG with a feature size of 8×8 removes most of the background once again. While PBAS has the lowest percentage of False Positives of all the algorithms, it also has the highest percentage of False Negatives. This trend was noticed in PBAS for many sequences. The authors of PBAS refer to it as 'Implicit Erosion Effect' [31]. Interestingly, the next best performance is obtained with ViBe and MGM-UM. Next best is RMoG with 1×1 fol-

lowed by Eigenbackground. Foreground definition for ViBe, MGM-UM and RMoG with 1×1 is very similar. The foreground boundary edge definition for RMoG with 8×8 is poorer once again. For the Boats sequence, RMoG with a feature size of 1×1 and PBAS produce the best results with RMoG having slightly higher False Positives and PBAS having slightly higher False Negatives. RMoG with 8×8 features also produce reasonably good results with the dynamic background only present in sporadic regions in the output. For the Canoe sequence, RMoG with a feature size of 1×1 produces the best output by a good margin with a low percentage of False Positives and False Negatives. While PBAS removes most of the background, it loses the foreground definition. The Fall sequence results are similar to the Waving Trees sequence with RMoG with 1×1 producing the best output again. RMoG with 8×8 handles the two Fountain sequences quite well with almost all the algorithms producing really good results particularly for Fountain2 sequence. All the algorithms have less than 5% of False Positives in this sequence. The Overpass sequence, with a tree swaying in the wind in the background, similar to the Waving Trees and Fall Sequences, has similar results to those two sequences with RMoG with 1×1 again coming out on top followed by PBAS.

The computational complexity of the different algorithms are also included in Table 1 for comparison. The complexity of the original MoG, as well as RMoG with r=8, and f=1 or f=8, can be computed from the expressions given in Table 1. The complexity of the Eigenbackground algorithm mainly depends on the PCA computation and this is given by $\mathcal{O}\left(N^2M + N^3\right)$ [34] where N is the total number of features per vector (or in this case, the total number of pixels in an image) and M is the number of images used in

the block computation of PCA. Note that this is a batch algorithm, hence the computational complexity is higher than the other algorithms. ViBe and PBAS are similar non-parametric modelling approaches, and the complexity for processing each pixel, without any post processing, is given by the time taken to search over the number of samples in the background model, which is given by B. Hence, the total complexity over the entire image for ViBe and PBAS is $\mathcal{O}(NB)$. Typically, the value of B was set around 20 and 35 in the original ViBe and PBAS papers respectively. The MGM-UM is very similar to the original MoG algorithm, and the asymptotic complexities are the same, given by $\mathcal{O}(NH(1+\log H))$. From these expressions, it can be noticed that the asymptotic complexities for all the algorithms, except Eigenbackground, are of the same order and comparable.

Summarising, it can be inferred that the RMoG algorithm performs quite well compared to the others, particularly in regard to background removal. Comparing columns 8 and 9, while the output of column 9 is less precise, it does benefit from the advantages of block based processing such as robustness to noise and background movement. The processing time for block based processing is also lesser compared to pixel based processing due to the reduction in the number of samples per frame as was seen in Section 5.1. The false positives due to the dynamic background are also reduced by a considerable amount. Furthermore, in real-world practical applications such as ours, the blocky outputs of the foreground detection are post-processed to provide a rectangular input, containing the original foreground object shape, to a human detection system. This eliminates the need to scan entire images for human signatures as the regions containing the foreground alone can be

locally processed. Therefore, the original shape of the human is retained for input to the human detection module. Isolated blocks of false positives can be easily filtered out by morphological processing. Block based processing is also found to be performed in background subtraction algorithms in the compressed domain. There, 8×8 blocks of DCT coefficients are used as features instead of pixel intensity values. Hence, these results (Column 9 of Figure 8) can be viewed as their spatial domain counterparts. The choice of different feature sizes also helps in obtaining suitable output based on the application. When a blocky result is sufficient, the results from the ninth column show that the dynamic regions in the background has been subtracted almost entirely compared to the other results. On the other hand, if a fine silhouette of the foreground is required, the results from the eighth column are on par, if not better than the existing state of the art methods. This flexibility is an important attribute of this algorithm.

6.5. CDNet 2014 Benchmark Evaluation

We further evaluated our algorithm on the entire ChangeDetection (CDNet 2014) benchmark dataset [35] consisting of fifty-three video sequences divided into eleven categories, each posing a particular surveillance challenge. Detailed results for individual video sequences can be found on the benchmark website [36] along with results for several other state-of-the-art algorithms. The quantitative results obtained with RMoG for each category, along with the average, are given in Table 4. The metrics in the first four columns of Table 4 are the average values for True Positive Rate, True Negative Rate, False Positive Rate and False Negative Rate. The fifth column is the Percentage of Wrong Classifications (PWC) or the error rate,

Category	Recall	Specificity	FPR	FNR	PWC	F-Measure	Precision
Baseline	0.7082	0.9981	0.0019	0.2918	1.5935	0.7848	0.9125
Camera Jitter	0.6669	0.9864	0.0136	0.3331	2.6794	0.7010	0.7605
Dynamic Background	0.7892	0.9978	0.0022	0.2108	0.4238	0.7352	0.7288
Intermittent Object Motion	0.4488	0.9950	0.0050	0.5512	4.6882	0.5431	0.8026
Shadow	0.6680	0.9936	0.0064	0.3320	2.1720	0.7212	0.8073
Thermal	0.3441	0.9991	0.0009	0.6559	5.1222	0.4788	0.9365
Bad Weather	0.5572	0.9991	0.0009	0.4428	0.8739	0.6826	0.8955
Low Framerate	0.5805	0.9922	0.0078	0.4195	1.6809	0.5312	0.5916
Night Videos	0.5524	0.9668	0.0332	0.4476	5.1606	0.4265	0.4345
Pan-Tilt-Zoom	0.6409	0.9278	0.0722	0.3591	7.4757	0.2470	0.2212
Turbulence	0.5780	0.9952	0.0048	0.4220	0.7314	0.4578	0.5701
Average	0.5940	0.9865	0.0135	0.4060	2.9638	0.5735	0.6965

Table 4: Quantitative Metrics for RMoG algorithm for each category in CDNet 2014 (Recall, Specificity, False Positive Rate, False Negative Rate, Percentage of Wrong Classifications, F-Measure, Precision)

which is given by the ratio of the falsely classified pixels to the total pixels used for evaluation, expressed as a percentage. The sixth column shows the F-Measure which is calculated as (2*Precision*Recall)/(Precision+Recall). The seventh column gives the average precision values. These correspond to the metrics used for evaluation on the CDNet website [36], enabling easy comparison. In addition, an example output is shown for each of the eleven categories in Fig. 9, along with their corresponding input frames and ground truth. (The grey areas in the ground truth are outside the region of interest, i.e., they are not considered when calculating the metrics.)

As we are particularly interested in dynamic backgrounds, we first discuss

the results for those categories that are most relevant to this scenario; Camera Jitter, Dynamic Background, Bad Weather and Turbulence. For the Camera Jitter category, the F-Measure is 0.7010 which is a 17% increase over the original MoG algorithm [36]. This also compares favourably to the top score of 0.7886 obtained by the CwisarDH algorithm [37]. A low error rate of 2.6794 in this category puts RMoG among the best algorithms and within 1% of the PWC obtained by CwisarDH. For this category RMoG is also ranked seventh, in specificity and precision, and eighth, in F-Measure, in comparison to the other algorithms in CDNet 2014 [36]. Camera jitter causes pixels across the scene to shake back and forth within a small region, thus region-based mixture modelling handles it quite well with very few false alarms, Fig. 9 (second row). Similarly for the Dynamic Background category, the F-Measure for the original MoG algorithm is 0.6330, whereas, for the RMoG algorithm, it is 0.7352, an increase of 16%. While this is comparatively lower than the top score of 0.8792 obtained by FTSG [38], it is still among the top six CDNet 2014 algorithms for F-Measure and top five for PWC. Figure 9 (third row) shows that for the overpass sequence, with waving trees and water in the back-ground, there are no false alarms, however, there are several false negatives within the detected foreground region. The Bad Weather category includes videos taken in challenging winter conditions like blizzards and snowfalls. While our RMoG algorithm has the second best specificity value in this category and is also among the best in terms of precision, the F-Measure is again brought down by the recall value. As can be seen in Fig. 9 (seventh row), there are no false positives in the blizzard scene, whereas there are false negatives present at the rear of the car. In the turbulence category,

the RMOG precision, 0.5701, is ranked sixth best in the benchmark. It is also able to remove most of the turbulent background, as evidenced by the high specificity value, 0.9952, and the output frame in Fig. 9 (row eleven).The results obtained for these dynamic-background related categories show that the RMoG algorithm is comparable to the state of the art in modelling and subtracting dynamic backgrounds in a scene.

Of the other categories in the benchmark, the Intermittent Object Motion category was one of the most challenging, as can be seen from its low recall, high PWC and low F-Measure. In these videos, a foreground object is left static in the scene for a long time period and then moved again. It is very difficult to differentiate these objects from the background after a short period of time. However, these results are still quite competitive with those of the CDNet 2014 benchmark, outperforming more than half the algorithms, with the PWC value still within in the top five algorithms. The RMoG algorithm is also capable of handling video sequences captured with low frame rate cameras reasonably well, as can be seen from the result in the eighth row of Fig. 9. Here the video was captured at 0.5 fps producing the effect of foreground objects jumping from one position to another between consecutive frames. In the Night Videos category, the bright headlights of the vehicles result in an increase in false positives, whilst a lack of proper illumination in other parts of the scene causes false negatives. The F-Measure of 0.4265 for RMoG is the seventh highest in this category among the CDNet 2014 algorithms [36]. Out of all the categories, the lowest F-Measure score and highest PWC for RMoG was obtained in the PTZ category, where the entire scene changes constantly, making it hard for the algorithm to adapt

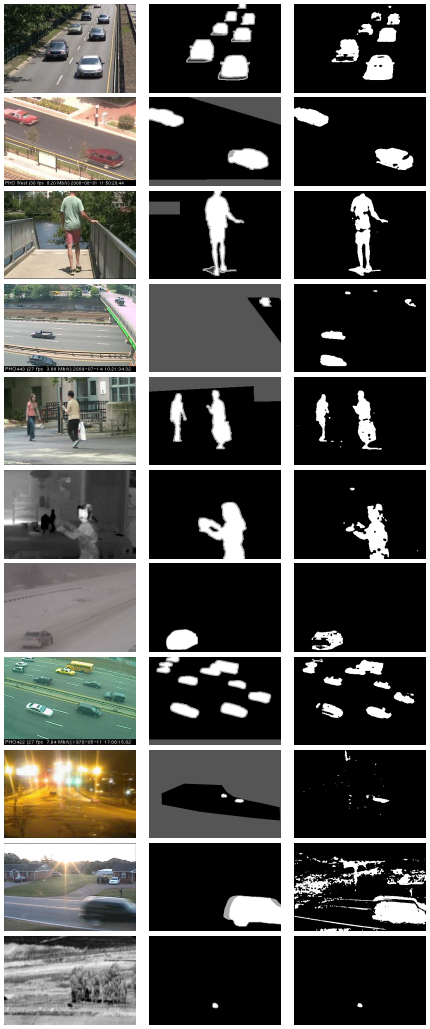


Figure 9: RMoG Sample Outputs for CDNet 2014 (First Column: Input Frame, Second Column: Ground Truth). First Row: Baseline (Highway); Second Row: Camera Jitter (Boulevard); Third Row: Dynamic Background (Overpass); Fourth Row: Intermittent Object Motion (Street Light); Fifth Row: Shadow (Backdoor); Sixth Row: Thermal (Dining Room); Seventh Row: Bad Weather (Blizzard); Eighth Row: Low Framerate (Turnpike-0.5fps); Ninth Row: Night Videos (Bridge Entry); Tenth Row: PTZ (Continuous Pan); Eleventh Row: Turbulence (Turbulence3)

to it on the fly, Fig. 9 (Tenth Row). However, it must be noted that a PWC of 7.5% for RMoG is still better than all but five algorithms in the benchmark [36]. We further noticed that RMoG did not perform very well in the Thermal category. While the precision is quite high, the recall is low. On further inspection, we found that since the images are essentially three identical channel intensity images, the distances between different pixel intensities are smaller, thus making it likelier for foreground regions to fall under the background model. In the Shadow category, while soft shadows are removed by the RMoG, hard shadow regions will invariably be detected as part of the foreground region, Fig. 9 (fifth row). However, this is the case in almost all background subtraction algorithms with no dedicated shadow removal component. These results suggest that the RMOG algorithm performs reasonably well in handling almost any type of challenge in surveillance videos.

Overall, the RMoG algorithm is one of the very best in reducing false positives, as is evident from the very high specificity values for all categories and an overall specificity value which is the best out of all the algorithms in the older CDNet 2012 benchmark and fifth best in the newer CDNet 2014 benchmark. RMoG also has the sixth lowest PWC value in the benchmark. This is due to the fact that RMoG models regions rather than individual pixels, thereby ensuring that dynamically moving pixel values are still associated with the correct background model component, provided the movement is local within a region. The benchmark website ranks the algorithms by calculating the rank across the seven different measures in all the categories and averaging them. This currently places the algorithm in the eighth position

in the CDNet 2014 benchmark [36].

As was the case in section 6.1, we observe that the recall value is relatively low for this algorithm. This is because of two main reasons. The first one is that stopped objects are quickly merged into the background, reducing the number of true positives over time. The second, and key, reason is that when a foreground pixel corrupts a background model, it immediately affects all the pixels in the surrounding region, thus rapidly classifying the whole region as background. This can be looked upon as the reverse effect of region-based background modelling where foreground regions are affected instead of the background regions. These are the two issues we intend to address in the future.

7. Summary and Conclusion

In summary, using the EM framework, we have formally derived model update equations for region based background modelling taking into consideration the relationship between pixels in a neighbourhood over time. We used these update equations to propose a background subtraction algorithm for foreground detection in highly dynamic scenes. We also generalised our algorithm to include different neighbourhood and feature size combinations. Our algorithm was then validated using various video sequences containing different types of dynamic background such as surface waves in water, trees swaying in wind and dynamic motion outside a moving bus. The results show that it is quite capable of handling scenes with complex, dynamic backgrounds effectively. By increasing the region size of the algorithm, the false positives due to the dynamic background are reduced compared to the

original modelling approach, while maintaining foreground definition. With larger feature sizes, while the foreground definition becomes blockier, the dynamic background is removed to an even greater degree. The effect of the learning rate when the feature size is equal to the region size is similar to that of original MoG however, when the region size is increased with respect to the feature size, the effect of the learning rate is diminished. The algorithm significantly improves upon the original MOG algorithm and is able to produce results similar and in many cases, even better, compared to four state-of-the art techniques. We further evaluated our algorithm on the exhaustive ChangeDetection (CDNet 2014) benchmark and showed that the model works very well in scenes having dynamic background and camera jitter, while also performing reasonably well in all other categories.

In conclusion, the proposed RMoG algorithm shows potential in relation to the robustness required for real-world deployment. In the future, we hope to optimise the region size based on the dynamics of the scene background. This will also help provide an insight into combining the benefits of both pixel-based and region-based modelling approaches depending on the scene context. Also, we will look into different strategies in order to prevent the foreground pixels from corrupting the background model. We believe that a separate foreground model, combined together with our robust region-based background model, will produce even better results across all categories.

Acknowledgements

We gratefully acknowledge the valuable comments made by the anonymous reviewers that helped us improve the paper significantly.

References

[1] R. Radke, S. Andra, O. Al-Kofahi, B. Roysam, Image change detection algorithms: a systematic survey, Image Processing, IEEE Transactions on 14 (2005) 294–307.

[2] Y. Benezeth, P.-M. Jodoin, B. Emile, H. Laurent, C. Rosenberger, Comparative study of background subtraction algorithms, Journal of Electronic Imaging 19 (2010).

[3] T. Bouwmans, Recent advanced statistical background modeling for foreground detection: A systematic survey, Recent Patents on Computer Science 4 (2011).

[4] S. Popa, D. Crookes, P. Miller, Hardware acceleration of background modeling in the compressed domain, Information Forensics and Security, IEEE Transactions on 8 (2013).

[5] C. Stauffer, W. E. L. Grimson, Adaptive background mixture models for real-time tracking, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1999, pp. 2246–2252.

[6] T. Bouwmans, F. E. Baf, B. Vachon, Background modeling using mixture of gaussians for foreground detection - a survey, Recent Patents on Computer Science 1 (2008) 219–237.

[7] G. Doretto, A. Chiuso, Y. N. Wu, S. Soatto, Dynamic textures, International Journal of Computer Vision 51 (2003) 91–109.

[8] P. Kaewtrakulpong, R. Bowden, An improved adaptive background mixture model for realtime tracking with shadow detection, in: In Proc. 2nd European Workshop on Advanced Video Based Surveillance Systems, 2001.

[9] Z. Zivkovic, Improved adaptive gaussian mixture model for background subtraction, in: International Conference on Pattern Recognition, 2004, pp. 28–31.

[10] H. Wang, P. Miller, Regularized online mixture of gaussians for background subtraction, in: 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2011, pp. 249–254.

[11] A. M. Elgammal, D. Harwood, L. S. Davis, Non-parametric model for background subtraction, in: European Conference on Computer Vision, 2000, pp. 751–767.

[12] A. Mittal, N. Paragios, Motion-based background subtraction using adaptive kernel density estimation, in: Internation Conference on Computer Vision and Pattern Recognition (CVPR), 2004, pp. 302–309.

[13] A. Tavakkoli, M. Nicolescu, G. Bebis, M. N. Nicolescu, Non-parametric statistical background modeling for efficient foreground region detection, Machine Vision Applications 20 (2009) 395–409.

[14] Y. Sheikh, M. Shah, Bayesian modeling of dynamic scenes for object detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2005) 1778–1792.

[15] P.-M. Jodoin, M. Mignotte, J. Konrad, Statistical background subtraction using spatial cues, IEEE Transactions on Circuits and Systems for Video Technology 17 (2007) 1758–1763.

[16] R. Yumiba, M. Miyoshi, H. Fujiyoshi, Moving object detection with background model based on spatio-temporal texture, in: Proceedings of the IEEE Workshop on Applications of Computer Vision, 2011, pp. 352–359.

[17] S. Zhang, H. Yao, S. Liu, Spatial-temporal nonparametric background subtraction in dynamic scenes, in: IEEE International Conference on Multimedia and Expo (ICME), 2009, pp. 518–521.

[18] G. Dalley, J. Migdal, W. E. L. Grimson, Background subtraction for temporally irregular dynamic textures, in: IEEE Workshop on Applications of Computer Vision, 2008, pp. 1–7.

[19] P. Dickinson, A. Hunter, K. Appiah, A spatially distributed model for foreground segmentation, Image Vision Computing 27 (2009) 1326–1335.

[20] S. Varadarajan, P. Miller, H. Zhou, Spatial mixture of gaussians for dynamic background modelling, in: 10th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2013, pp. 63 – 68.

[21] C. M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), 1 ed., Springer, 2007.

[22] L. Bottou, Online algorithms and stochastic approximations, in: Online Learning and Neural Networks, Cambridge University Press, 1998.

[23] X. Fang, W. Xiong, B. Hu, L. Wang, A moving object detection algorithm based on color information, Journal of Physics: Conference Series 48 (2006) 384.

[24] J. Zhong, S. Sclaroff, Segmenting foreground objects from a dynamic textured background via a robust kalman filter, in: Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on, 2003, pp. 44–50.

[25] L. Li, W. Huang, I. Y. H. Gu, Q. Tian, Foreground object detection from videos containing complex background, in: Proceedings of the eleventh ACM international conference on Multimedia, 2003, pp. 2–10.

[26] K. Toyama, J. Krumm, B. Brumitt, B. Meyers, Wallflower: Principles and practice of background maintenance, in: Seventh International Conference on Computer Vision, 1999, pp. 255–261.

[27] N. Goyette, P. Jodoin, F. Porikli, J. Konrad, P. Ishwar, Changedetection.net: A new change detection benchmark dataset, in: Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on, 2012, pp. 1–8.

[28] N. Oliver, B. Rosario, A. Pentland, A bayesian computer vision system for modeling human interactions, Pattern Analysis and Machine Intelligence, IEEE Transactions on 22 (2000) 831–843.

[29] O. Barnich, M. Van Droogenbroeck, Vibe: A universal background subtraction algorithm for video sequences, Image Processing, IEEE Transactions on 20 (2011) 1709–1724.

[30] F. Baf, T. Bouwmans, B. Vachon, Type-2 fuzzy mixture of gaussians model: Application to background modeling, in: Proceedings of the 4th International Symposium on Advances in Visual Computing, ISVC '08, Springer-Verlag, 2008, pp. 772–781.

[31] M. Hofmann, P. Tiefenbacher, G. Rigoll, Background segmentation with feedback: The pixel-based adaptive segmenter, in: Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on, 2012, pp. 38–43.

[32] Vibe, `http://www2.ulg.ac.be/telecom/research/vibe/`, 2013.

[33] A. Sobral, BGSLibrary: A opencv c++ background subtraction library, `http://code.google.com/p/bgslibrary/`, 2013.

[34] Q. Du, J. E. Fowler, Low-complexity principal component analysis for hyperspectral image compression, Int. J. High Perform. Comput. Appl. 22 (2008) 438–448.

[35] Y. Wang, P.-M. Jodoin, F. Porikli, J. Konrad, Y. Benezeth, P. Ishwar, Cdnet 2014: An expanded change detection benchmark dataset, in: Change Detection (CDW 2014) at CVPR 2014, 2014 IEEE Workshop on, 2014, pp. 387–394.

[36] Cdnet, `http://www.changedetection.net`, 2014.

[37] M. De Gregorio, M. Giordano, Change detection with weightless neural networks, 2014.

[38] R. Wang, F. Bunyak, G. Seetharaman, K. Palaniappan, Static and moving object detection using flux tensor with split gaussian models, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2014.